UNIVERSIDAD TECNOLÓGICA METROPOLITANA
FACULTAD DE INGENIERÍA
ESCUELA DE INFORMÁTICA

INF-628
Prof. Mauro Castillo V.

PRUEBA N°1 ESTRUCTURA DE DATOS
SEGUNDO SEMESTRE 2006

NOMBRE:..... SECCIÓN:.....

PARTE I: Marcar la opción que corresponda (3*buenas - malas)

1. Dada la siguiente declarativa

```
class Uno{
        public:
     int funcion(int x) { return x+2;}
};
Uno u;
```

Cual es la opción correcta?
a) int i = funcion(2);
b) cout << u.funcion(2) << endl;
c) float u.funcion(2) = 3.5;
d) Ninguna de las Anteriores.

2. Dada la siguiente declarativa:

```
class Ejemplo
{       public:
                int prueba(int x) { return x * m + n;}
                int n;
        private:
                int m;
};
Ejemplo obj;
```

Que sentencia es correcta?
a) cout << obj.m;
b) obj.n = 10;
c) obj.m = obj.prueba(2);
d) Ninguna de las Anteriores.

3. Qué valor se imprime?

```
int i = 3, j = 6;
float r = 7.0;
int cosa(int t, int *a) { return t * (*a);}
int cosa(float t, int *a) { return t + *a; }
int cosa(double t, int *a) { return t / *a;}
cout << cosa(r, &i);
```

a) 21
b) 10
c) 2.33
d) Ninguna de las Anteriores.

4. Se dispone de la siguiente declaración

```
template <class T, class U>
T funcion(U p)
{
        return (T) p;
}
```

Son correctas?

```
I ) int x = funcion(3.2);
II ) float r = funcion(3);
III ) cout << funcion(2);
```

a) Solo I y III
b) Solo II
c) Solo III
d) Ninguna de las Anteriores.

5. Dada la siguiente declarativa

```
Class Numero
{       public:
                int doble();
        private:
                int base;
}
```

Que implementación es la correcta para definir la función miembro "doble".
a) int doble() { return base + base}
b) int Numero::doble() { return base + base}
c) Numero doble() { return base + base}
d) Ninguna de las Anteriores.

PARTE II: (15 puntos)
Se dispone de la clase Complejo:

```
class Complejo
{
        public:
                Complejo(float r, float i);
                void imprimir();
                Complejo conjugado();
                Complejo suma(Complejo x, Complejo y);
                Complejo resta(Complejo x, Complejo y);
                Complejo invertir();
        private:
                float real, imag;
};
```

Se pide escribir la función miembro *invertir*, que permite obtener el inverso de un número complejo:
el inverso de $(a + bi) = (a - bi)/(a^2 + b^2)$

PARTE III (15 puntos)

Se pide escribir la función externa **Division(X,Y)** que permita dividir dos objetos de tipo *Fraccion*.

```
class Fraccion{
        public:
                Fraccion(int n=0, int d=1) : num(n), den(d) {}
                void Lee(int &n, int &d){ n=num; d=den;}
                void Modifica(int n, int d) { num=n; den=d;}
                int Numerador() {return num;}
                int Denominador() {return den;}
        private:
                int num, den;
};

int main()
{
        Fraccion B(5), C(2,3), A;
        A = Division(B,C);
        cout << "B dividido por C = " << A.Numerador() << "/" << A.Denominador() << endl;
}
```

PARTE IV (15 puntos)
Dada la siguiente definición de la clase Polinomio:

```
class Polinomiostruct{
        public:
                Polinomiostruct(int grado = 0);
                ~Polinomiostruct();
                void Llenar();
                float& Coef(int i);
                float Evaluar(float x);
                void Imprimir();
                void Derivar();
        private:
                struct{
                        int mGrado;
                        float mCoef[100];
                }mEsd;
};
```

Se pide escribir la función miembro *Derivar* que permite derivar un polinomio dado.

PAUTA I

1) b
2) b
3) b
4) d
5) b

PAUTA II

```
Complejo Complejo::invertir()
{
        Complejo temporal(real/(real*real+imag*imag), (-1)*imag /(real*real+imag*imag));

        return temporal;
}
```

PAUTA III

```
Fraccion Division(Fraccion F, Fraccion R){
        int n =  F.Numerador() * R.Denominador();
        int d = F.Denominador() * R.Numerador();
        Fraccion X(n,d);
        return X;
}
```

PAUTA IV

```
void Polinomiostruct::Derivar()
{
        for(int i = 1; i <= mEsd.mGrado; i++) {
               mEsd.mCoef[i-1] = mEsd.mCoef[i] * i;
        }
        mEsd.mGrado = mEsd.mGrado - 1;
}
```